Performance evaluation and simulation of peer-to-peer protocols for Massively Multiplayer Online Games

Performance Evaluation and Simulation of Peer-to-Peer Protocols for Massively Multiplayer Online Games

Lu Liu[1], Andrew Jones[1], Nick Antonopoulos[1], Zhijun Ding[2], Yongzhao Zhan[3]

[1]School of Computing and Mathematics, University of Derby, Derby, Derbyshire, DE22 1GB, United Kingdom
[2]Department of Computer Science and Technology, Tongji University, China
[3]School of Computer Science and Telecommunication Engineering, Jiangsu University, Jiangsu, China

Corresponding Author: Dr Lu Liu

School of Computing and Mathematics, University of Derby

Derby, Derbyshire, DE22 1GB

United Kingdom

Email: l.liu@derby.ac.uk

Tel: +44 (0) 332 59 1707

Fax: +44 (0) 332 59 7741

Abstract

Massively Multiplayer Online Games are networked games that allow a large number of people to play together. Classically MMOG worlds are hosted on many powerful servers and players that move around the world are passed from server to server as they pass through the environment. Running a large number of servers can be challenging and there are many considerations for a developer who wants to create a game to enter the MMOG market. If it is possible to use a P2P network to host an MMOG successfully, the costs of running a server farm are taken out of the equation. This will allow for groups with small budgets to enter the MMOG market and help competition in the market place. In this paper, the methods for the design of P2P massively multiplayer game protocols have been presented. Performance bottlenecks have been evaluated and highlighted by using simulations. The business viability has also been discussed in this paper.

Keywords

MMOG; P2P; Networking; Simulation

1. Introduction

Massively Multiplayer Online Games (MMOG) are networked games that allow a large number of people to play together at the same time. The aim of many MMOG developers is to allow the largest number of people possible to interact with each other in the game world. This aim could not be achievable without advanced network strategies such as multi-server networking architectures. There are many considerations for a developer who wants to create a game to enter the MMOG market. Latency and synchronisation are the most important issues in multiplayer gaming. Most real-time multiplayer games depend on all the players witnessing events at exactly the same time so that all players have a chance to interact with the games mechanics. Another challenge is how to host the virtual environment that the players will be playing in. Classically MMOG worlds are hosted on many powerful servers and players that move around the world are passed from server to server as they pass through the environment.

Massively Multiplayer Online Games have been around since September 1996 when Meridian 59 was released. Since then, MMOGs have become incredibly popular. The most well-known MMOG today, World of Warcraft, currently boasts about 12 million subscribed accounts [1].

Games like World of Warcraft have far too many players to host on one server. Even with a system of servers working together to host all the game clients there is a point where the overhead becomes too great. Many MMOGs on the market do not have scalable server architectures. It is because of this that MMOGs divide their player base and put them on different server clusters. In the game Everquest 2, each server can host around 200 people and their multi-server approach can host a total load of around 10000 to 15000 players in one go [2]. For a game with so many subscribed users this means that the developer should run a very large number of servers to make sure that all their players can play the game whenever they want.

Running a large number of servers can be challenging. To begin with, the company needs to buy a large amount of hardware and employ many staff to keep the hardware running. The energy cost is high and the price of the large amount of floor space that these servers take up is also considerable. As a game developer it is hard to justify these costs, especially in view of the narrowing margins in the industry. Many developers have turned to subscription based business models to make sure that if they are forced to continue to host the servers for a long time, they will have income that is proportional to the number of necessary servers.

Peer-to-peer (P2P) networks have been used extensively in other areas of computer networking in order to get rid of the need for any one person to host a large farm. Many very popular applications such as Skype use P2P techniques so that the users host the communications rather than the company having to host a central server.

In P2P networks each computer can act as both the client and the server. This differs from a traditional client/server architecture where machines can act either as a server or a client but not both. There are also two types of P2P network. Pure P2P networks are networks that are run completely by peers; if a peer was removed then the network would continue to function. In contrast, hybrid P2P networks are networks that contain one or more central servers as well as the peers. The network is defined as hybrid if these central servers are vital to keep the network running [3].

There are many advantages and disadvantages of P2P networks over the standard client/server model. In normal client/server architecture the bandwidth that the server has available is limited. In P2P networks better load balancing can be achieved where a whole resource can be sourced from many different servers where, at each, a part of that resource already exists. Many P2P networks split large files into “blocks” that can then be distributed around the network. Once all the blocks of a file have been retrieved from the network, the peer can then assemble the whole file. Many games companies make use of P2P networks for patch and update distribution in-order to alleviate the strain on their server’s bandwidth. Riot Games, whose main game has over 15 million registered players [4], uses Pando Media Booster (Pando Networks) to improve the speed at which their users can obtain patches and updates for their game. The disadvantage of this delivery system is that it is the customer’s bandwidth that is used and their computing power is being for the companies’ purposes. In a world where people are limited to the amount of uploading and downloading they are allowed by their internet service provider, this is not ideal for customers.

As well as information that should be sent between clients and servers, another important part of a game server’s job is to do calculations based on what players are doing within the virtual world. In the Server/Client model, the amount of calculation that can be done is limited by the server hardware. In a pure P2P network the limit is that of the hardware of all the peers put together. The calculation capacity of a hybrid P2P network is only limited by the hardware of all the peers and the central server put together. A successful P2P protocol for gaming could use the computational power of the peers efficiently so that the network and the amount of calculation increases proportionally as peers join the network and the available computational power goes up [5].

If it is possible to use a P2P network to host an MMOG successfully, the costs of running a server farm are taken out of the equation. This will allow for groups with small budgets to enter the MMOG market. This would help competition in the market place as well as allowing for less budget-focused games to enter the market. Without the need to make large sums of money to keep afloat, experimentation and creativity can be more freely used within the MMOG medium.

Small games studios that want to get into the Massively Multiplayer Online Game industry can use the information in this paper to make an educated decision as to whether or not to use P2P technologies in their game development. It is hoped that the analysis of different protocols in this paper should allow game studios to decide whether or not a P2P MMOG is a viable idea and, if so, which protocol would be best for them based on their needs. This also indicates where future research needs to focus to improve the business viability of P2P MMOGs.

To the best of our knowledge, there is no systematic evaluation of the performance and suitability of P2P protocols for MMOG. The aim of this paper is to analyse current P2P protocols for MMOG and to compare them to each other in order to assess their viability for business applications and their effectiveness in reducing ‘bottlenecks’. This is important due to the large volume of P2P communications required to run such games without central servers.

The objectives are to implement a range of the most recently proposed massively multiplayer online game P2P protocols and to simulate a large scale network using a P2P simulator. The network data in areas key to real-time MMOGs will be collected and the data from the different protocols will be compared to deduce which best handles the possible performance bottlenecks. The analysis of the collected data will be used to discuss whether there might be a more optimal approach to P2P MMOG protocols that has not yet been covered by current research. Using this discussion the aim is to direct where future research might concentrate in order to optimise further the key performance bottlenecks.

This paper contains the following: Section 2 gives an overview of P2P systems and P2P protocols for games. The P2P protocols for MMOGs are analysed in Section 3. Section 4 details the methodology of a series of evaluation and simulations. In Section 5 the results are presented and analysed. Sections 6 and 7 discuss the performance of the protocols for MMOGs and reach conclusion about the state of the current protocols and what this means for the games industry.

2. Related work

2.1 P2P Systems

Most of today's Internet applications are distributed by using a client-server architecture, such as WWW, FTP, email, etc. However, the centralisation of the client-server architecture raises a series of issues which are caused by the limitation of resources at the server side, such as network bandwidth, CPU capability, and storage space. In contrast to the client-server paradigm, the P2P architecture does not rely on a centralised server to provide services, which offers an appealing alternative to the client-server model especially for large-scale distributed applications. In the P2P model, each peer node acts as both client and server, requesting resources from as well as routing queries and serving resources for other peer nodes. A P2P network is a logical overlay network over a physical infrastructure, which provides a virtual environment for P2P developers to easily design and implement their own communication environment and protocols on the top of existing networks [6].

As a new design pattern, P2P has been widely used in the design of popular Internet applications, especially for file sharing, Internet Telephony and online gaming. An explosive increase in the popularity of P2P file sharing applications has been witnessed by millions of Internet users [7]. P2P file sharing has become one of the most popular Internet activities. Today's popular P2P file-sharing applications, such as BitTorrent and Kazaa, have more than one million users each at any point of time [8]. P2P Internet telephony is another quickly developing area based on P2P techniques. Skype is a very popular VoIP P2P application, often with tens of millions users online. Skype not only provides PC-to-PC communication services, but also offers PC-to-phone communication services, phone-to-PC communication services and PC-to-PC video conferencing services. Unlike open VoIP protocols like SIP, IAX, or H.323, the Skype protocol requires no servers and is able to work across different types of network connections including firewalls and NAT [9]. In addition to file sharing and Internet telephony, P2P gaming is also increasing popular with millions of players are connected to several online gaming communities. P2P gaming is the focus of this paper which will be further analysed and evaluated in the following sections.

2.2 P2P for games

In order for a games studio to make decisions about how their P2P protocol will work, they should think through what areas could be optimised the most in order to create a smooth experience for the players. According to research, game state management and latency are the two main existing issues on P2P gaming which needs to take into account [10]. A game is characterized by a set of states while the states are modified along the game course by events. The P2P game system needs to guarantee that the game state is consistent among all players. Latency is another challenges faced by P2P gaming. Delays should be kept beneath the human perceptible threshold which is typically 200ms according to Distributed Interactive Simulation (DIS) and the High Level Architecture (HLA) standards [15]. If delay is above this figure, P2P game players may become uncomfortable. Moreover, game systems also need to guarantee the same delay for all players. Otherwise, there is unfairness especially for the users experiencing a longer delay.

In order to solve these issues from effecting player's experience, gameplay trick can be employed to reduce the effect of latency. As Fritsch's paper suggests [2], one such trick is for the clients to make an educated guess about where other players are going to be during the next update so that if communication between the two clients is slow, the effects of the lag are not shown to the user. This can create a much smoother experience for all the peers but can cause issues like 'rubberbanding' where players stop running in the direction the game was expecting them to and an avatar jump from one location to another in an unrealistic way.

Managing the amount of game state information to pass around the network is vital in making a scalable P2P gaming protocol. Fan, Trinder and Taylor define these design choices as 'interest management' [11]. Many interest management algorithms have been researched [12].

How messages traverse the network is another critical feature of P2P MMOG protocols. Connections can be either Unicast or Multicast. Unicast approaches allow low latency between the connected peers but raise the bandwidth requirement. Multicast approaches help cut the bandwidth usage but this is dependent on the structure that is used to decide what peers the messages will be sent to [11].

3. P2P Protocols for MMOG

3.1 N-Tree

This N-Tree protocol [13] focuses on propagation of events around the network. The protocol uses an N-Tree to manage peers based on their area of interest. N-Trees are a tree data structure where each internal node has N dimensions. Therefore, if N were three then you would have an octree like the ones used in computer graphics [14]. As explained below, the result of using this approach means that players will be communicating with other players that are closer to them first in-order to cut down on latency. This therefore maintains synchronisation between them and close peers.

Players trigger events by interacting in the game world. These events are labelled with the area that this event will impact. The N-Tree sends the event to all the peers who are within the area that the event affects. Most events will only affect the peers in close proximity in the game world and thus messages will only have to be exchanged within one group of leaves. If events are dealt with by the peers in the order they received them, there is already a suitable event ordering system in place where more important events to the peer are transferred faster than events that happen further away and therefore have less importance.

Nodes in the N-Tree are either parent nodes or leaves (nodes with no children). Some peers in the network are selected to manage the nodes and leaves of the N-Tree. All peers in the network are leaves in the tree. Each group of leaves is a group that sends events to each other using multicast. There is an event whose scope is greater than the group of leaves. Then it is passed through leaders of groups to higher level nodes until the scope of the event is reached and then the event is propagated down the tree to all the leaves on nodes below it. Nodes have a location within the world and peers are managed by the closest node to them. If a peer's location is between two nodes then they are assigned as leaves on both nodes. Nodes are dynamically located based on player location in the world and more nodes are created when the threshold number of children is exceeded. The threshold is defined as 2^N in the N-Tree. In latency-dependant game types like first-person shoots a low threshold will need to be used to lower the amount of communication that needs to occur.

A distributed hash table is used to discover peers in the N-Tree. Peers register their IPs and their location in the game world so that when a new peer wants to join the network they can get a list of peers that are in the same area as them. A join message is sent to one of the peers in the list and that peer has to see whether or not the new peer is in its area of interest. If it is not, it will send the join message up or down the tree. When the join message reaches a peer that is within its scope the new peer joins the tree at this location. If this new peer pushes the children of the node over the threshold, a divide message is sent to the peers. The virtual world space that this node was representing is then split evenly between the N dimensions of the tree. The leaders of the new nodes are determined and notify the parent of their leadership status.

When a player wants to log out and leave the network, a leave message is sent to all the leaves in its group. If the peer is the leader, a new leader will be selected from the group and the parent node is notified. If a player leaves the network without sending a leave message due to a disconnection or game crash then the peer's disappearance should be handled by the parent when it tries to send messages to the peer and realises it is not able to communicate. In the case that the group finds its leader has disconnected, they can elect a new leader and send information to the parent but if a parent finds that the leader has disconnected then it should consult a list of members of the group that is sent periodically so the parent can inform the group and get a new leader elected. While a new leader is being elected the parent should queue events so that state management is maintained, although this may cause some lag to occur. When peers leave the network the parent should make a decision whether to collapse the sub-tree the node was in or not.

The sub-tree can be collapsed if there are less than or equal to t/2 leaves where t is the leaf threshold. These steps mean that an optimal tree is always maintained.

3.2 QuOn

QuOn [15] is a protocol based on a quad-tree in a similar way to N-Tree. QuOn is developed with player behaviour in mind. QuOn uses a system where partitioning is localised to each peer in the system. All messages that need to be sent in the QuOn protocol are unicast directly between two players. This means that latency depends on the routing in the network between these two players only, rather than a routing overhead around the overlay. The system uses an aura based area of interest approach. Therefore, players connect to all other players inside their own area of interest. Events from outside the area of interest are sent and received through ’binding neighbours’. Neighbour classification and management is handled using a point quad-tree that is unique to each player and is stored locally. The quad-tree contains the positions of all the known neighbours. When a neighbour is added to the tree, the region containing the new neighbour is split into four sub-regions at that position. Communication is optimised for each player individually.

Neighbours are classified into three groups. Each group ensures game state synchronicity based on their location. Players whose location resides in the area of interest of the player are known as ’direct neighbours’. If a neighbour is a direct neighbour of the player then it is true that that the player is a direct neighbour of the neighbour. The second group of neighbours are classified as ’binding neighbours’. Binding neighbours have the job of relaying important state information around the world. Within top level of a player’s local quad-tree, the world will be split into four quadrants. One binding neighbour is selected as the closest neighbour based on world location for each one of the four regions. It is very possible that the binding neighbours are also direct neighbours since closer neighbours are preferred. All neighbour detection is done through these binding neighbours. ’Soft-state neighbours’ are needed to make sure those players who need to interact across larger distance than the area of interest can do so. Soft-state neighbours are only maintained while interaction between these players is ongoing. The quad-tree and the neighbour classifications should be remade each time a player moves or receives a movement update but, since the trees are local, this does not cause network overhead.

The problem with this network structure is that network failure is not considered. In order to keep the network running, a peer should store a list of old binding neighbours to use as backups if the current binding neighbours are lost.

In order to join the network, the new player sends a message to any player within the network. This peer compares its location to that of the new peer and sends the peer’s join message to the closest known peer to the new player’s location in the peer’s local tree. This is done recursively until the closest player to the new player is found. The new player’s tree is then built using all the information from the closest player. This system does mean that, as more players join the network, the joining time goes up linearly.

When leaving the network, peers will send a leave message to all its neighbours with a list of its known neighbours so that other peers can rebuild their trees based on the new information. If a player disconnects unclearly, players who have this player as a neighbour will be unable to communicate with the player. A new tree will be built based on the neighbours that they know about.

3.3 VAST

VAST is a protocol that is based on a Voronoi diagram as described by Guibas and Stolfi [16]. An aura-based area of interest is used. The solution described is to split the world into regions that are dynamic and that move with the user.

A Voronoi diagram of the world is made by splitting the world into tessellating regions that have one player in each region. Using the Voronoi diagram, neighbours can be classified in a similar way to the QuOn protocol. Neighbours that share a border with the region are classified as enclosing neighbours. Neighbours that are in regions that are within the player's area of interest are boundary neighbours. Each player has a local copy of the Voronoi diagram that it will maintain. Unicast connections will exist between all the neighbours meaning that close players will have more chance of being synchronised and have lower latency. Whenever a player sends movement messages, updates to all its neighbour peers inform them peers about possible new neighbours and the Voronoi diagrams are recalculated.

When a player wants to join the network a join message is sent to an arbitrary peer who then forwards the message on the a peers whose position is closer to the joining player until the peer is found that is hosting the region the new player would reside in. When this region's host is found they will get a list of neighbours from the host and then new Voronoi diagrams need to be calculated. It is important to remember that this will be done on each player's local machine and there is no more communication needed than with a simple move command.

In order to leave the network the peer should notify all of its neighbours. All the neighbours that receive the notification will remove the peer from their list of neighbours and update their Voronoi diagram. Since Voronoi diagrams are recalculated, the P2P overlay is consistently maintained.

4. Evaluation and Simulations

In order to further analyse the strengths and weaknesses of these current P2P protocols, simulations of the implementations of these protocols need to be conducted and relevant data recorded, so that their performance can be compared. An analysis of the current protocols may show areas that need to be improved as well as showing which methods are succeeding. This paper focuses on finding the best existing methodology and building on it so that a more complete and scalable protocol can be designed.

To test a proposed MMOG protocol, it needs to be implemented and run in a network environment. The problems with testing a large multiuser network by programming the game then sitting hundreds of users down at computers are fairly clear. Having to develop a product to a testing stage without any sign that it will work well enough to be used in a commercial product does not make sense economically, let alone do the cost and logistics of such a large test. When a system is very large, simulation is really the only viable option. In academia, simulation of P2P networks is commonplace with almost all research papers that propose protocols such as QuOn [13], VAST [15] and NTree [16] having results of multiple simulations displayed as graphs.

In order to simulate a P2P protocol, the protocol should be implemented and be used by software that acts as if it were a real node within a real network. The software should realistically simulate latency and connection issues as though it were a real network node.

There are many P2P network simulators available to academics such as ScaleNet[17], P2PSim [18] and PlanetSim [19]. "Towards yet another peer-to-peer simulator" [20] surveys P2P networking papers and discusses why there are so many different P2P simulators. They suggest that key functionality is missing from a large number of the simulators and that no one simulator has been built that caters for everyone's needs. Custom built simulators are very common with 62% of the papers that the authors [20] looked at using a custom built simulator. This does cause some difficulty in verifying the quality of the results that are presented in a paper.

The simulator that seems to be most used in the field of game protocols appears to be OverSim [21]. This is predominantly because Oversim was developed at the Institute of Telematics, Universität Karlsruhe alongside research for P2P protocols like QuOn[13]. Since the protocols that are involved in the analysis have all been simulated in OverSim in the past, it is the natural choice for the current work.

OverSim has been developed to be a modular system where the network underlays, overlays and the application that is being simulated can be plugged in and out. It is made using the OMNeT++ open source simulation framework that uses a discrete event simulation system. A great thing about OMNeT++ is that it has built-in GUI displays for network topologies, nodes and messages.

OverSim has three underlay network models that can be used to simulate with: Simple, SingleHost, and INET. The SingleHost underlay model emulates a single host, which can be connected to other single host emulations across a network like a LAN or the internet. The INET underlay model is made using the INET framework that is in OMNeT++ which simulates all network layers and can be useful but it is fairly CPU intensive. Thus, it is not very useful for the simulation of a large network like that needed for an MMOG.

The simple network underlay is a model where data packets are sent from one overlay node to another using a routing table. Delay is simulated between the nodes based on the distance between them. Distance between nodes is simulated by placing them in a large two-dimensional space at random. Each node is accessed using network characteristics based on real world data such as bandwidth, access delay and packet loss. The simple underlay is set up perfectly for quick simulation of large networks which makes it very useful for MMOG protocol research.

What makes OverSim the best option for MMOG protocol research is the overlay protocols that communicate with the underlay network via at UDP/IP interface. Many overlay protocols are already implemented such as QuOn [13]. New overlays can be implemented easily because the authors have identified functions that many overlay protocols implement and allow the overlay developer to implement them without changing the entire system. For instance, a generic lookup function can be edited in order to use new lookup functionality.

OverSim uses the Common API for P2P overlays [22] to allow developers to write application simulators. OverSim provides an application called SimpleGameClient that simulates players on game clients either walking around randomly or walking around in groups.

In order to be able to analyse the current P2P protocols, simulations have been run that test all three of the protocols on the most important performance bottlenecks. Data from the simulations will be recorded and compared against one another to see where strengths and weaknesses may lie.

It is important to note that although these simulations will give realistic results for bandwidth usage and connectivity for the network layer there are many other things that might need to be considered in a full scale game product so the resulting figures should not be used as definitive values to prove whether or not P2P MMOGs are practical.

5. Results & Analysis

In order to gauge the performance of the P2P networks in different scenarios, two sets of simulations were run. To test how the protocols would cope with handling different world sizes, a set of simulations was run that maintained a set target node count of 512 players. 512 players were chosen because this is roughly the number of players that are in the busiest zone on a highly population server in World of Warcraft at peak times. The players of MMORPGs expect to be able to play in a large world and the protocols should be scalable and not become a problem at high area sizes. Since the amount of communication between clients will go up as more players exist within each other’s area of interest it should be advantageous to have the players spread across a large map. In order to simulate players the OverSim module ’SimpleGameClient’ was used to send movement messages to the overlays simulating players walking randomly around the game area. The movement in all the simulations was updated 6 times a second. In addition to the player’s movement behaviour, churn is active in the simulations so that players are joining and leaving the network as the simulation goes on. In all the simulations the area of interest of the protocols was set to 50 metres.

5.1 Effect of world dimension

The graph in Figure 1 shows the average rate of download the node used to receive all the messages it required to maintain itself for different world dimension settings.

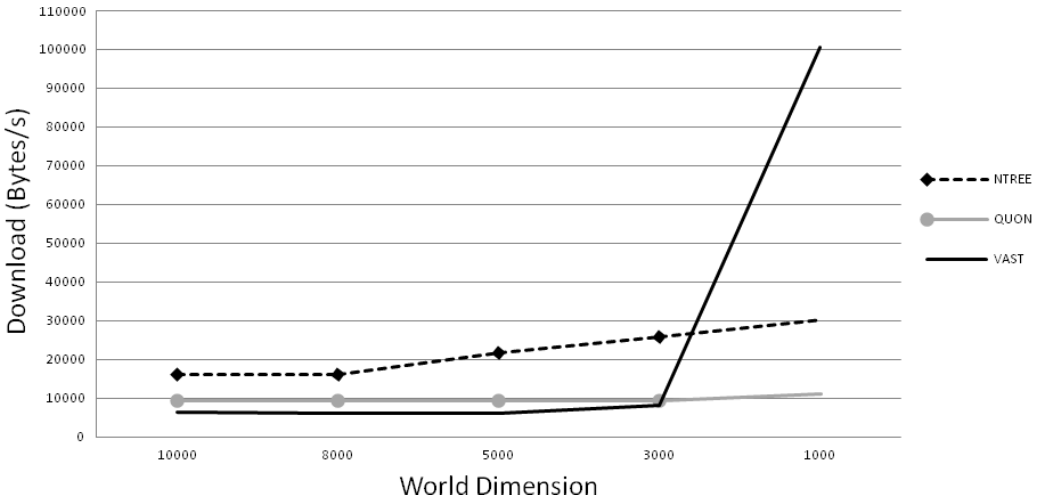

Figure 1: A graph showing the average downloaded bytes per second in simulations with different world dimensions.

All the messages received have been sent from other nodes in the network and it is unsurprising that the upload mirrors the download somewhat. It is important though to remember that these two figures cannot simply be regarded together as ’bandwidth’ because download and upload are treated differently by internet service providers in different countries. In the UK, 10 megabits per second internet packages are commonplace but the upload speeds for such a connection rarely go over 1 megabit per second. Figure 2 shows the average bytes per second sent from a node to the rest of the network.

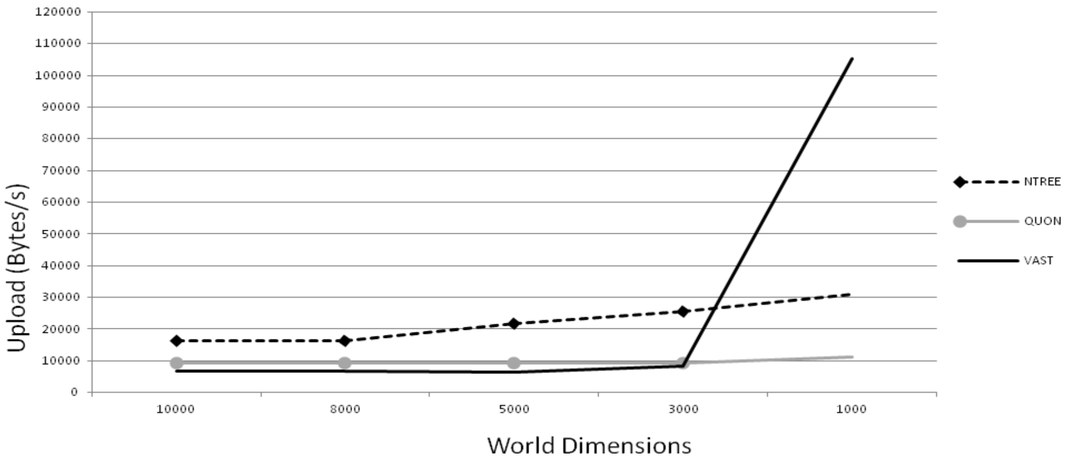

Figure 2: A graph showing the average uploaded bytes per second in simulations with different world dimensions.

As well as the bottleneck of bandwidth, the other bottleneck that has been discussed is latency. In order to show the latency that would directly affect the players in the network, the simulations monitored the movement delay. The average delay is presented in Figure 3.

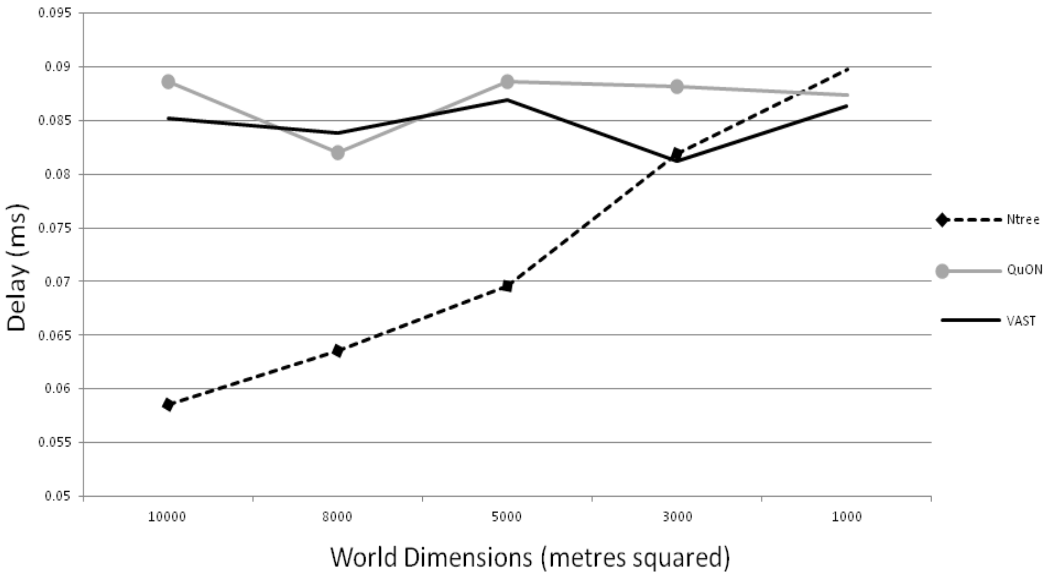

Figure 3: A graph showing the movement delay in milliseconds in simulations with different world dimensions.

Figure 1 shows the average downloaded bytes per second in simulations. The download rate shows the size of the received messages that were used to keep a client synchronised in the network and all the movement messages from the nodes around it.

With the players spread out across a large area of 10,000 by 10,000 metres squared it is the VAST protocol that uses the least bandwidth for downloading of network messages. When using VAST's area of interest system to have connections between neighbour peers it is advantageous to have as few as possible people within that area of interest. There will be a considerably fewer boundary and enclosing nodes when 512 players are spread over such a large area. The difference in downloaded bytes does not go up in the 8,000 by 8,000 or 5,000 by 5,000 dimensions, perhaps indicating that the number of enclosing and boundary neighbours is still relatively low. When 512 players are evenly distributed around a 5,000 by 5,000 metre world there is and density of 0.1024 players per square metre. This means that on average, since the players are moving randomly in the simulations, there will be around 4 other players in the area of interest at any one point. When the density goes up to 0.17 players per square metre in the 3000 by 3000 metre area there is a slight increase in neighbour nodes and the download requirement rises to near 10,000 bytes per second. When the world dimensions are as low as 1,000 by 1,000, the population density goes up to 0.512 players per square metre. This means that in a 1,000 by 1000 area there will be roughly 25 neighbours in the area of interest. The extra messages that need to be sent in an area of this density have a dramatic effect on the download rate needed to support the protocol. VAST really fails to scale well past a certain level of density.

N-Tree starts as the protocol with the large bandwidth requirement at 10,000 by 10,000 and the bandwidth requirement slowly rises as the area becomes smaller. Each node in the tree needs to download information from its children and its parent and the number of connections will be maintained whatever the density in a network of this size. The rise in required download rate can be attributed to the height of the tree and the changing of the tree structure. Event propagation in a large, sparsely populated area requires messages to be sent to all the nodes around the node in the network since the closer nodes are in the virtual world, the closer they will be in the network. As the area becomes smaller there are more tiers of the tree within the leaf nodes area of interest. This means that even locally scoped events like movement should be conveyed up many tiers of the tree and then all the way back down, propagated by multiple peers, to get to every node within the scope.

QuOn starts at just under 10,000 bytes per second and manages to maintain that bandwidth throughout the simulations until a slight rise in the most densely populated zone simulation. The reason that the bandwidth stayed so constant throughout the simulations is that in QuOn the connections are limited to a maximum of 4 binding neighbours. The only communication needed outside of those binding neighbours is with the soft-state neighbours, to make sure the binding members are correct (a fairly low cost overhead), and to the direct neighbours that are the other nodes in the area of interest. It is important to note though that it is the 4 binding members communication that makes the QuOn protocol more bandwidth intensive in the low population density zone simulation.

QuOn comes out on top of scalability in these simulations. However, it is VAST that causes the lowest bandwidth usage in a low density area. N-Tree's bandwidth usage is fairly low but the consistent rise shows issues in extremely crowded game areas.

Figure 2 shows the average uploaded bytes per second in simulations. Uploaded bytes represent the size of the messages that were sent from the node to keep the nodes around it in the network and send all the movement messages of the nodes client to the node around it in the network. As noted earlier, the protocol's upload bandwidth mirrors the download bandwidth because everything one node is sending, another node is receiving. It is important to think about this bandwidth usage from an upload speed point of view however, since internet service providers treat it differently. Since it not possible to get 1 to 1, download to upload connections in many countries, upload will be the bottleneck before download bandwidth becomes an issue.

Figure 3 shows the average movement delay experienced by the clients in the network. This is the delay that the player will experience between someone moving near to them in the virtual world and their client reacting to that movement.

For both QuOn and VAST there is no clear change in delay between the 10,000 by 10,000 area and the 1000 by 1000 area. Since both protocols maintain direct contact with the nodes within their area of interest, the latency between the nodes stays roughly same. Looking at the latency difference between the 10,000 by 10,000 area and the 1000 by 1000 area for N-Tree, though, there is a noticeable difference. N-Tree attempts to keep latency low by having nodes close to each other in the network if they are close to each other in the virtual world. This works when the nodes are spread out around the 10,000 by 10,000 area and nodes should only communicate between nodes with the same parent. When more than one group of nodes is in each other's area of interest, however, there is some delay as messages should travel through multiple peers.

In terms of delay, QuOn and Vast seem to have reasonably small delay times consistently. N-Tree shows that, at the levels of population density in the simulations, it does have lower delay times but does show a weakness in its ability to scale to densely populated areas. With N-Tree there will come a point where the bandwidth usage becomes too great. It is not unrealistic in MMOGs to have 512 players in the same area and with N-Tree this would cause an unacceptable level of bandwidth usage that would cause all the players to lag.

5.2 crowded area test

All the protocols seem to handle large areas well but as the area in which players are gathered becomes smaller, bandwidth and latency issues become more pronounced. In many games there are areas such as cities or battlegrounds where crowed activity is far more common and areas such as dungeons where forming groups is actively encouraged. While this is good for gameplay, developers should be aware of how this might affect the latency in these areas.

The next set of results show networks of varying player numbers in a set area of 1,000 by 1,000 metres. These dimensions were chosen to emulate the established games' highest player density areas. Once again OverSim's SimpleGameClient was used to emulate players walking randomly around the area.

Upload and download were both measured in the simulations to produce Figures 4 and 5.

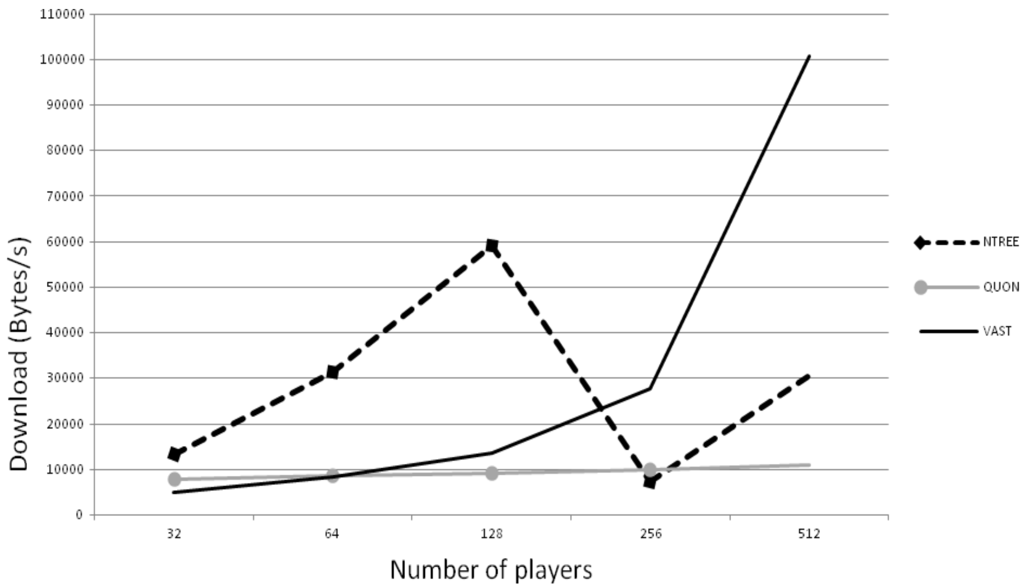

Figure 4: A graph showing the average downloaded bytes per second in simulations with numbers of players.

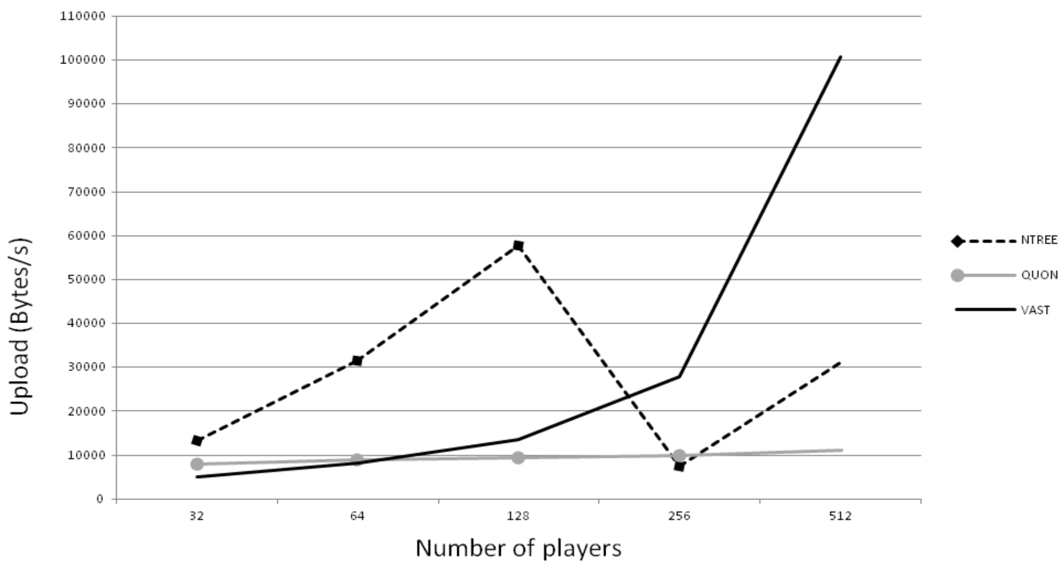

Figure 5: A graph showing the average uploaded bytes per second in simulations with numbers of players.

The reason the bandwidth used plummets with 256 players and 512 players is not that the protocol becomes more efficient at this level of player density. During the simulation the protocol was unable to cope with the churn and the player density and the whole network broke down. Figure 7 shows erroneous nodes across the simulation of 512 players in a 1000 by 1000 metre area. While there might be a way to engineer backup nodes to stop this kind of breakdown from occurring, it would cost more overhead bandwidth usage. The breakdown shows a real lack of robustness in the protocol.

As well as the bandwidth measurements, the movement delay was also measured in the simulations of 1000 by 1000 m areas. Figure 6 shows the change in movement delay between simulations of different densities.

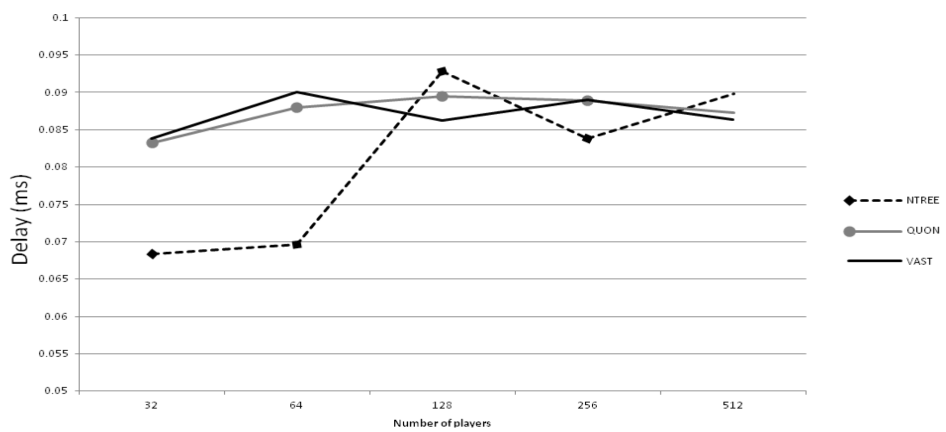

Figure 6: A graph showing the movement delay in milliseconds in simulations with different numbers of players.

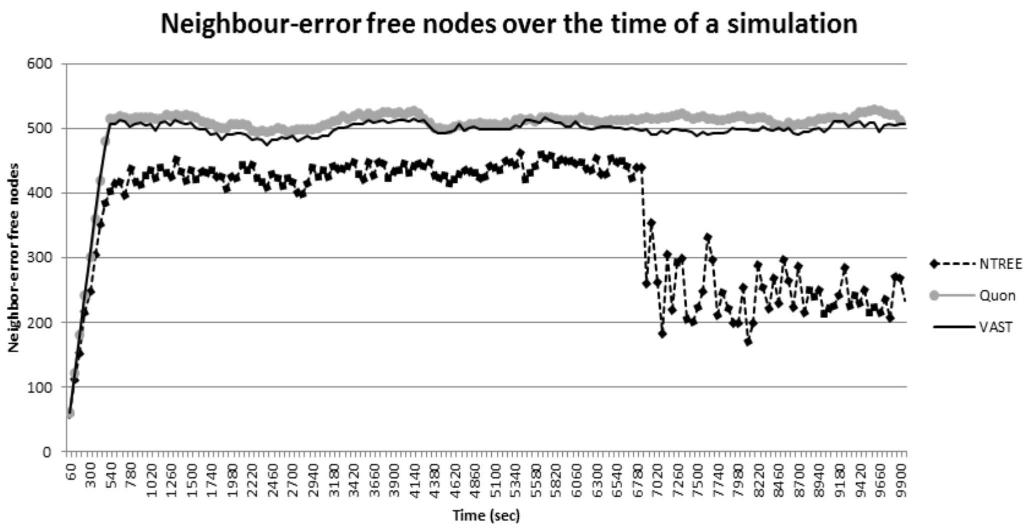

Figure 7: A graph showing the neighbour-error free nodes over the time in simulations of 512 people in a 1000 by 1000 metre area.

Figure 4 shows the average downloaded bytes per second in simulations of a set area of 1000 by 1000 metres. The change in download rate in relation to the number of players in the world shows how scalable the protocols are when dealing with crowed areas of a virtual world. In order to be viable for business, download rates should stay low enough in the most crowed area of the virtual world to still be viable for consumers.

As shown in the area dimension test, QuOn scales well with the population density and maintains low bandwidth requirements in all densities. VAST does a good job in the low population density of 32 and 64 players but quickly degrades when the number of players in the protocols area of interest goes up.

The results of N-Tree in the crowded area tests are not very good. As shown in the area dimension simulations, N-Tree does not scale well when peers are in close proximity since communications should traverse multiple tree tiers. This coupled with the churn (nodes leaving and joining the network) caused messages to go missing and caused a breakdown of the network. Somewhere in between 128 and 256 players in a 1000 by 1000 metre area the N-Tree protocol becomes very unstable as even locally scoped messages are passed through many peers making the chance of message loss increase substantially.

Figure 5 shows the average upload rate in simulations of a set area of 1000 by 1000 metres. This represents the data needed to be sent from a node to the nodes around it in order to maintain the network and distribute the movement messages of the nodes around them in the network.

Just as in the previous set of simulations, the upload mirrors the download. Since upload is the most likely bottleneck, though, VAST should be criticised as, in the most crowed area of the simulation, its required download rates are ten times that of the QuOn protocol.

Figure 6 shows the average movement delay experienced by the clients in the network. It is important that the protocol can deal with the most crowded area in the game without the players feeling restricted by the latency. If the delay on movement is too high it will break immersion for the players and cause frustration.

Between 32 and 128 players the N-Tree protocol's delay rises the same way it did in the previous simulation (Figure 3). After 128 players, though, the results cannot be trusted as the network suffered a large breakdown. Delay in the QuOn and VAST networks is kept to a minimum using direct connection. Both networks sacrifice bandwidth in order to keep delay low.

6. Discussion

The bandwidth requirements of the protocols in the simulations range from 4.88 kilobytes per second to 98.36 kilobytes per second for both upload and download. While the UK's average broadband speed is as high as 6.2Mbit/s (793.6 kilobytes per second) the upload speeds are a lot lower [23]. Only one ISP provides upload speeds higher than 1.5Mbit/s (192 kilobytes per second). With upload being the bottleneck here, we can see that all the protocol's upload rates fit within the UKs average upload rates but it is important to note that other factors could have an effect on the bandwidth requirements. Some ISPs will control the bandwidth so that when uploading occurs, the bandwidth for downloading is substantially lowered. Other network traffic such as VOIP software and web browsers may be used in tandem with the game or there may be more than one computer on the local network that wishes to play the game at the same time causing heavy bandwidth demand at the same time. Other software in the game itself will undoubtedly cause more bandwidth usage. The bandwidth used by the simulations only indicates the bandwidth required to maintain the network and send movement messages. Many other event messages would have to be sent based on how complex the game is. As well as other events in the game, most MMOGs allow the players to have a high level of persistence in regards to their character between play sessions. These protocols do not accommodate this and thus a secondary solution needs to be developed. If the game were to have the database of characters distributed around the network as well, it would cause a massive bandwidth overhead that would make the levels of upload currently designated to the protocol untenable. A hybrid approach to MMOG with dedicated servers for user data is perhaps a little more realistic with current bandwidth levels although this does not go all the way towards satisfying the need for no server farm.

The delay of VAST and QuOn is always going to be low as long as there is enough bandwidth for the protocols to use. N-Tree, on the other hand, does not seem to scale too well. If the problems that cause the network breakdown seen in the second set of simulations were to be fixed, there would still be the problem of delay in the N-Tree protocol as the amount of people crowded into an area increases.

7. Conclusion

From the evaluation results obtained from this paper, we can see that N-Tree and VAST do not scale as well as QuOn. It is the set number of direct connections that QuOn uses that give it an edge. While VAST would work well in a game where players are distributed around the world, this is not practical because players of MMOGs are drawn to each other as player interaction is what sets MMOGs apart from other game genres.

More research needs to be done into reducing the overheads of maintaining the network so that game events can be sent while still keeping the bandwidth at reasonable levels. Research like this is on-going but there is a serious lack of research into how the protocols can be turned into a network middleware that would support all the required features like data storage, load balancing, cheat detection and support distributed AI.

For a company to start producing a P2P MMOG today, it would be a fairly premature decision. On the other hand it is a brilliant time for the creators of network game middleware to produce tools for companies since, not only will networking middleware companies already have the staff with the correct expertise, but also developing the P2P element of a MMOG and a game at the same time would take far too many resources for even the most successful game studios.

Acknowledgment

The work reported in this paper has been supported by the Sino-UK Higher Education Research Partnership for PhD Studies, Natural Science Foundation of Jiangsu Province of China, RLTF e-learning Cloud Programme, National Natural Science Foundation of China Program (61272074), China 973 Fundamental R&D Program (2011CB302600) and Visiting Research Fellow Program of Tongji University.

References

[1] Blizzard Entertainment. (2005) World of Warcraft Subscriber base reaches 12 million worldwide. Available at: http://us.blizzard.com/en-us/company/press/pressreleases.html?id=2847881. Accessed 10 July 2013

[2] Fritsch, T., Ritter, H. and Schiller, J. (2005) The Effect of Latency and Network Limitations on MMORPGs: A Field Study of Everquest2. In Proceedings of 4th ACM SIGCOMM Workshop on Network and system support for games, pp 1-9. doi: 10.1145/1103599.1103623

[3] Schollmeier, R. (2001) A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications. In Proceedings of the 1st International Conference on Peer-to-Peer Computing. pp 101-102. doi: 10.1109/P2P.2001.990434

[4] Riot Games. (2011) Riot Games Celebrate the end of league of legends season one: Registered players surpass 15 million, with over 1.4 million players every day. Available at: http://www.riotgames.com/news/riot-games-celebrates-end-league-legends-season-one-registered-players-surpass-15-million-over. Access 10 July 2013

[5] Fischbach, K., Schoder, D. and Schmitt, C. (2005) Core Concepts in Peer-to-Peer Networking. Chapter 1, Peer-to-peer Computing: The Evolution of a Disruptive Technology, Idea Group Publishing.

[6] Liu L., Antonopoulos, N, Mackin, S. (2007) Fault-tolerant Peer-to-Peer Search on Small-World Networks, Future Generation Computer Systems, Elsevier Science, 23: 921-931. doi: 10.1016/j.future.2007.03.002

[7] Zhao S., Stutzbach D., and Rejaie R. (2006) Characterizing Files in the Modern Gnutella Network: a Measurement Study, in Proceedings of SPIE/ACM. Multimedia Computing and Networking. pp 1-13.

[8] Liu L., Antonopoulos N.,, Mackin S., Xu J. Russell D. (2009) Efficient Resource Discovery in Self-organized Unstructured Peer-to-Peer Networks, Concurrency and Computation: Practice and Experience, Wiley, 23:159-183. doi: 10.1002/cpe.1329

[9] Baset, S.A. and Schulzrinne, H.G. (2006) An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol, in Proceedings of 25th IEEE International Conference on Computer Communications. pp 1-11. doi: 10.1109/INFOCOM.2006.312

[10] Neumann C., Prigent N., Varvello M., Suh K. (2007) Challenges in Peer-to-Peer Gaming, ACM SIGCOMM Computer Communication Review, 37: 79-82.

[11] Fan L., Trinder P., and Taylor H. (2010) Design Issues for Peer-to-Peer Massively Multiplayer Online Games. International Journal of Advanced Media and Communication. 4:108-125. Inderscience Publishers, Geneva, Switzeland. doi: 10.1504/IJAMC.2010.032138

[12] Boulanger J.-S., Kienzle J., and Verbrugge C., (2006) Comparing Interest Management Algorithms for Massively Multiplayer Games. In Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games. doi: 10.1145/1230040.1230069

[13] GauthierDickey C., Lo V. and Zappala D. (2005) Using N-Trees for Scalable Event Ordering in Peer-to-Peer Games. Proceedings of the international workshop on Network and operating systems support for digital audio and video. pp 87-92. doi: 10.1145/1065983.1066005

[14] Foley J. D., Van Dam A., Feiner S. K., and Hughes J. F. (1996) Computer Graphics: Principles and Practice, Addison-Wesley, Boston, USA.

[15] Krause, S. Backhaus, H. (2010) QuON – a Quad-Tree Based Overlay Protocol for Distributed Virtual Worlds. International Journal of Advanced Media and Communication, 4:126-139. doi: 10.1504/IJAMC.2010.032139

[15] Guibas L., Stolfi J. (1985) Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi Diagrams. ACM Transactions on Graphics (TOG), 4: 74-123. doi: 10.1145/282918.282923

[17] Siebert, M., Xu, B., Banniza, T., Keller, R., Dekorsy, A., Eichinger, J., Bless, R., Baumgart, I., and Stefanov, S. (2006) ScaleNet - Converged Networks of the Future. Themenheft “IP basierte mobile Systeme”, vol. 5, pp.253–263.

[18] Li, J., Stribling, J., Morris, R., Kaashoek, M. and Gil, T. (2005) A performance vs. cost framework for evaluating DHT design tradeoffs under churn. In Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies. vol. 1, pp. 225–236.

[19] Garca, P., Pairot, C., Mondjar, R., Pujol, J., Tejedor, H. and Rallo, R. (2005) Planetsim: A new overlay network simulation framework. Software Engineering and Middleware, 3437:123–136.

[20] Naicken, S., Basu, A., Livingston, B., Rodhetbhai S. and Wakeman, I. (2006) Towards yet another peer-to-peer simulator. In Proceedings of the Fourth International Working Conference Performance Modelling and Evaluation of Heterogeneous Networks (HET-NETs '06). In: HET-NETS 06, Ilkley, UK.

[21] Baumgart, I., Heep, B. and Krause, S. (2007) OverSim: A flexible overlay network simulation framework. In Proceedings of the 10th IEEE Global Internet Symposium (GI '07) in conjunction with IEEE INFOCOM 2007, Anchorage, AK, USA.

[22] Dabek, F., Zhao, B., Druschel, P., Kubiatowicz , J. and Stoica, I. (2003) Towards a common api for structured peer-to-peer overlays. In Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03), 2735: 33–44.

[23] Ofcom. (2011) Average broadband speed is still less than half advertised speed. Available at: http://media.ofcom.org.uk/2011/03/02/average-broadband-speed-is-still-less-than-half-advertised-speed/. Accessed 10 July 2013